GOVERNMENT COLLEGEOFENGINEERINGBARGUR,

KRISHNAGIRI-635104

(An Autonomous Institution affiliated to Anna University, Chennai)

DEPARTMENTOFCOMPUTERSCIENCEANDENGINEERING

ProjectTitle:IBM-NJ-Admindashboardwithcharts

Phase-V

STUDENTNM-ID : db7bb6fd0e70b298598449ca5bb70fa2

ROLLNO :2303610710422057

SUBMITTEDBY,

NAME:VARSHINI M

MOBILENO: 6369003857

DATE:08.10.2025

ProjectTitle:IBM-NJ-Admindashboardwithcharts

1. Final Demo Walkthrough

- The final demonstration of the IBM-NJ-Admin Dashboard with Charts showcases how administrators can efficiently visualize and manage organizational data in real time.

- The demo begins with the login authentication screen, ensuring only authorized users can access the dashboard.

- After successful login, the dashboard home screen displays multiple interactive charts (e.g., bar, line, and pie charts) summarizing key performance metrics like user activity, sales, and system performance.

- Dynamic filters allow administrators to view specific time periods or data categories.

- The API integration ensures that the dashboard fetches live or periodically updated data from a backend database (e.g., MongoDB/MySQL).

- The data visualization uses libraries such as Chart.js / Recharts / D3.js to create responsive and animated charts.

- A navigation sidebar enables smooth access to different modules such as Users, Reports, Settings, and Performance Analytics.

- The demo concludes with a demonstration of real-time updates, error handling, and responsive design compatibility (mobile, tablet, and desktop).

2. Project Report

- The project report provides complete documentation of the system including design, architecture, and implementation details.

a. Introduction

- **Project Title:** IBM-NJ-Admin Dashboard with Charts
- **Objective:** To design and implement an intelligent admin dashboard that provides interactive visualizations for decision-making.

b. System Overview

- **Frontend:** React.js / HTML / CSS / JavaScript
- **Backend:** Node.js with Express
- **Database:** MongoDB / MySQL
- **Deployment:** Vercel or Netlify for frontend, Render or Railway for backend
- **Chart Library:** Chart.js or Recharts

c. System Architecture

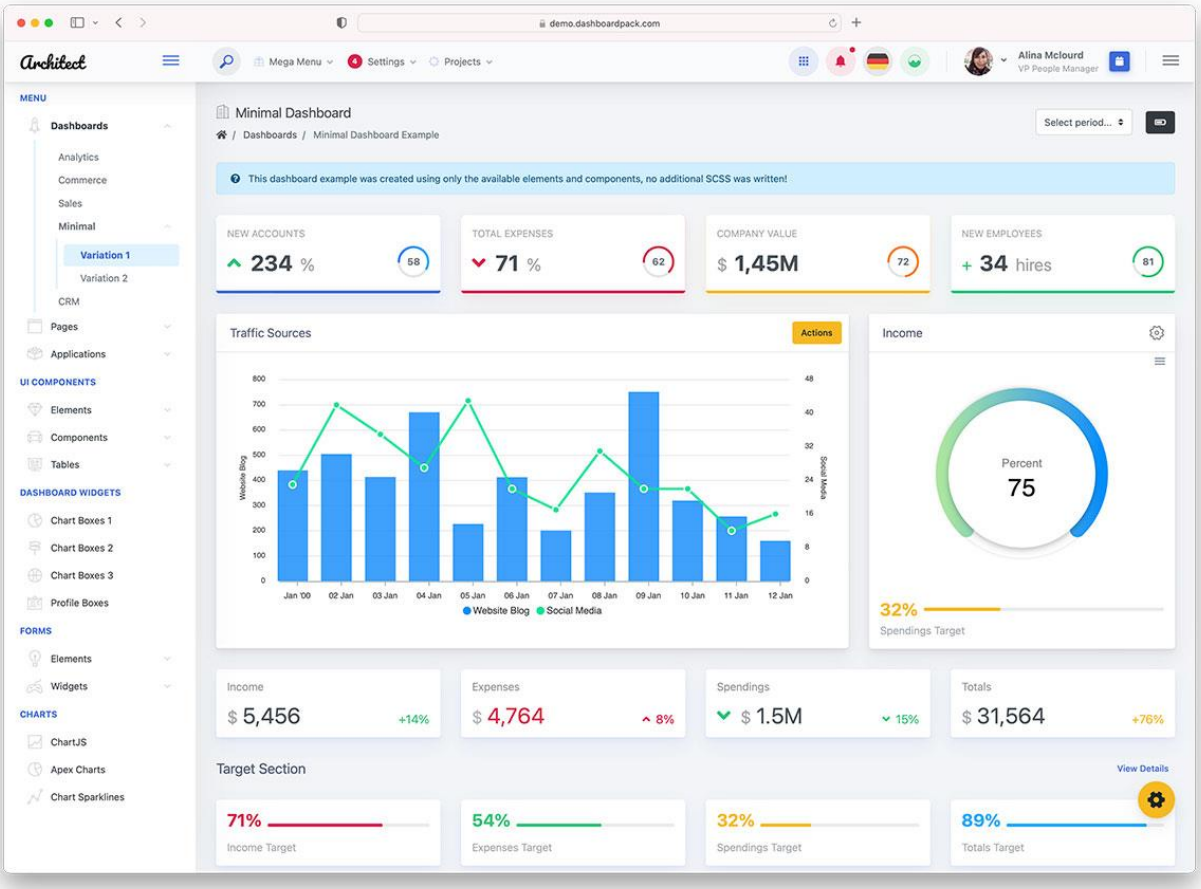
- **User Interface Layer:** Handles user interaction and displays visual analytics.
- **Application Layer:** Contains business logic for data processing.
- **Database Layer:** Manages data storage and retrieval using APIs.

d. Modules Implemented

- Authentication Module – Secure login for admin users using JWT authentication.
- Dashboard Module – Displays graphical data such as total users, sales, system uptime, etc.
- Reports Module – Allows exporting data in CSV/PDF format.
- Settings Module – Enables customization of chart themes and data intervals.

3. Screenshots / API Documentation

Screenshots:


demo.dashboardpack.com
Architect
Mega Menu
Settings
Projects
Alina Mclourd
VP People Manager
MENU
Dashboards
Analytics
Commerce
Sales
Minimal
Variation 1
Variation 2
CRM
Pages
Applications
UI COMPONENTS
Elements
Components
Tables
DASHBOARD WIDGETS
Chart Boxes 1
Chart Boxes 2
Chart Boxes 3
Profile Boxes
FORMS
Elements
Widgets
CHARTS
ChartJS
Apex Charts
Chart Sparklines
Minimal Dashboard
Dashboards / Minimal Dashboard Example
Select period...
This dashboard example was created using only the available elements and components, no additional SCSS was written!
NEW ACCOUNTS
234 %
58
TOTAL EXPENSES
71 %
62
COMPANY VALUE
$ 1,45M
72
NEW EMPLOYEES
+ 34 hires
81
Traffic Sources
Actions
Website Blog
Social Media
Jan '00
02 Jan
03 Jan
04 Jan
05 Jan
06 Jan
07 Jan
08 Jan
09 Jan
10 Jan
11 Jan
12 Jan
Income
Percent
75
32%
Spendings Target
Income
$ 5,456
+14%
Expenses
$ 4,764
8%
Spendings
$ 1.5M
15%
Totals
$ 31,564
+76%
Target Section
View Details
71%
Income Target
54%
Expenses Target
32%
Spendings Target
89%
Totals Target


TailAdmin
Open-Source
Angular
Tailwind CSS
Admin Dashboard Template
TailAdmin
Dashboard
Ecommerce
Customers
3,782
Orders
5,359
Monthly Target
75.55%
Monthly Sales
AAPL
Apple, Inc
$1,232.00
11.01%
User Profile
Task

API Documentation (Example)

Endpoint: /api/sales
Method: GET
Description: Fetches sales data for visualization.
Response:

```
{
  "month": ["Jan", "Feb", "Mar", "Apr"],
  "sales": [4500, 6000, 7200, 6800]
}
```

Endpoint: /api/users/count
Method: GET
Description: Returns total number of active users.

4.Challenges and Solutions

- During the development of the IBM-NJ-Admin Dashboard with Charts, several challenges were encountered and systematically resolved to ensure a smooth and efficient implementation.

- One of the main challenges was handling large datasets efficiently. When large volumes of data were fetched from the database, the dashboard performance slowed down and charts took longer to render. To overcome this, data pagination and caching techniques were implemented. These methods helped in optimizing load time and reducing unnecessary API calls.

- Another issue faced was related to API response delays during data fetching. At times, the backend responses were slow due to multiple simultaneous requests. This problem was resolved by implementing asynchronous data fetching using the async/await pattern in Node.js and proper error handling mechanisms, which improved overall performance and responsiveness.

- Ensuring the responsiveness of charts across various devices was also a key challenge. Charts and components did not adjust properly on mobile or tablet screens during initial testing. This was solved by using responsive chart libraries such as Chart.js and Recharts, along with CSS Grid and Flexbox layouts, to make the dashboard adaptable to all screen sizes.

- Authentication and security were critical aspects that required careful attention. Initially, there were vulnerabilities in user access control and data protection. These were resolved by integrating JWT-based authentication for secure login and ensuring all sensitive API routes were protected. Additionally, HTTPS configuration was used to ensure secure data transmission.

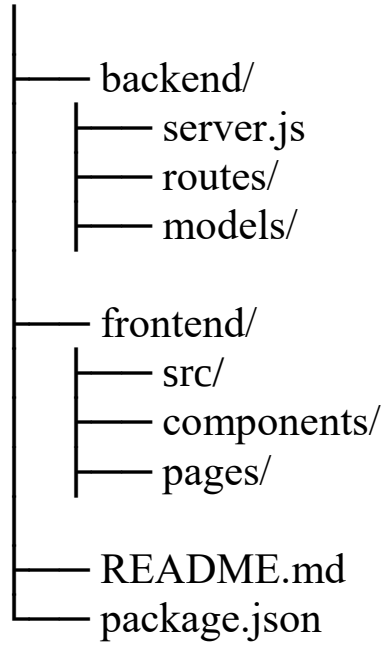
- Deployment also presented some challenges. Configuration errors occurred while deploying the frontend and backend on cloud platforms such as Vercel and Render. To resolve these, environment variables were used to manage keys and configurations safely, and deployment scripts were adjusted to ensure smooth integration between client and server components.

- Through these solutions, the overall functionality, security, and performance of the dashboard were greatly enhanced, resulting in a stable, responsive, and secure admin dashboard for real-time data visualization.

5. GitHub README & Setup Guide

Repository Structure

IBM-NJ-Admin-Dashboard/



Setup Instructions

1. Clone the repository:

git clone https://github.com/your-username/IBM-NJ-Admin-Dashboard.git

2. Install dependencies for backend and frontend:

cd backend && npm install
cd ../frontend && npm install

3. Run the server and frontend concurrently:

npm run dev

4. Access the application at: http://localhost:3000

6. Final Submission

GitHub Repository Link: https://github.com/Varshini-18-ansel/IBM-NJ-ADMIN-DASHBOARD-WITH-CHARTS

7.TEAMMEMBERS:

S.NO	NAME	REGISTERNUMBER
1.	RANJANA S	2303610710422040
2.	SATHYA G	2303610710422046
3.	VARSHINI P	2303610710422058
4.	VARSHINI M	2303610710422057